C# and .Net Development

Mr Rizvi Learning Log

Basic Outputs and Inputs of Strings

.NET Editor

```
1  Console.Write("Congratulations!");
2  Console.Write(" ");
3  Console.Write("You wrote your first lines of code.");
4  Console.WriteLine();
5  Console.WriteLine("This line starts on a new line :) ");
```

Output

```
Congratulations! You wrote your first lines of code.
This line starts on a new line :)
```

String Data Type

```
string name = "Bob";
int messages = 3;
decimal temperature = 34.4m;

Console.Write("Hello, ");
Console.Write(name);
Console.Write("! You have ");
Console.Write(messages);
Console.Write(" messages in your inbox. The temperature is ");
Console.Write(temperature);
Console.Write(" celsius.");
```

Float , numbered followed by F /f.

```
Hello, Bob! You have 3 messages in your inbox. The temperature is 34.4 celsius.
```

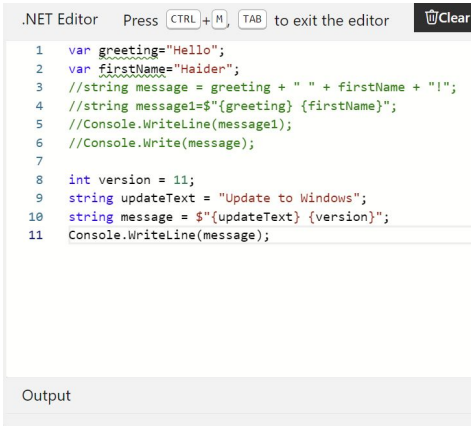
String Data Type

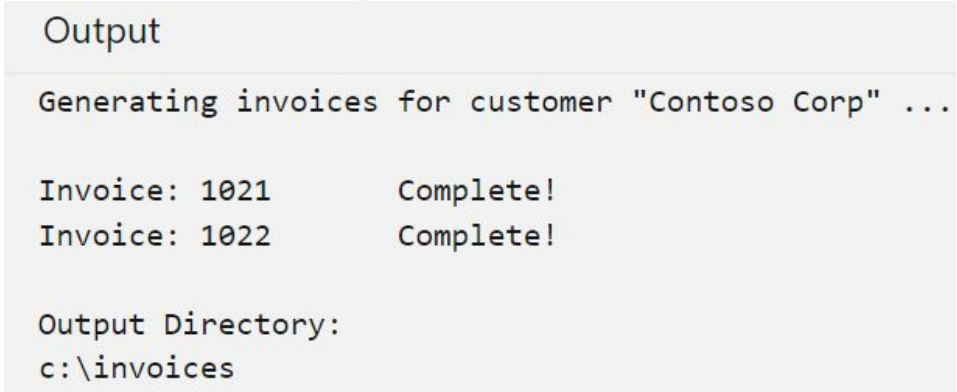
```
var message;
```

Implicitly typed local variable uses var, it sets data type for variable based on the data that is being assigned.

String Manipulation

```
1   Console.WriteLine("Generating invoices for customer \"Contoso Corp\
2   Console.WriteLine("Invoice: 1021\t\tComplete!");
3   Console.WriteLine("Invoice: 1022\t\tComplete!");
4   Console.WriteLine("\nOutput Directory:\t");
5   Console.Write(@"c:\invoices");
6
7   // To generate Japanese invoices:
8   // Nihon no seikyū-sho o seisei suru ni wa:
9   Console.Write("\n\u65e5\u672c\u306e\u8acb\u6c42\u66f8\u3092\u751f\u
10  // User command to run an application
11  Console.WriteLine(@"c:\invoices\app.exe -j");
```





Mathematical Operations + and concatenation

```
1   int firstNumber = 12;
2   int secondNumber = 7;
3   Console.WriteLine(firstNumber + secondNumber);
4   string firstName = "Bob";
5   int widgetsSold = 7;
6   Console.WriteLine(firstName + " sold " + (widgetsSold+7) + " widget
7   Console.WriteLine(firstName + " sold "+ widgetsSold + 7 + " widges"
```

Output

```
19
Bob sold 14 widgets.
Bob sold 77 widges
```

.NET Editor Press CTRL+M, TAB to exit the

```
1   int sum = 7 + 5;
2   int difference = 7 - 5;
3   int product = 7 * 5;
4   int quotient = 7 / 5;
5
6   Console.WriteLine("Sum: " + sum);
7   Console.WriteLine("Difference: " + difference);
8   Console.WriteLine("Product: " + product);
9   Console.WriteLine("Quotient: " + quotient);
```

```
decimal decimalQuotient = 7.0m / 5;
Console.WriteLine($"Decimal quotient: {decimalQuotient}");
Console.WriteLine("Decimal quotient:" + decimalQuotient);
// concatanated strings on line 3 and used dollar
```

Mathematical Operators(Decimals)

```
1   decimal decimalQuotient = 7 / 5.0m;
2   decimal decimalQuotient2 = 7.0m / 5.0m;
3   Console.WriteLine(decimalQuotient);
4   Console.WriteLine(decimalQuotient2);
```

```
1   decimal first2 = 7;
2   int first=7;
3   int second = 5;
4   decimal quotient1=(decimal)first /(decimal)second;
5   var quotient2 = (decimal)first /second;
6   var quotient3=first/(decimal)second;
7   var quotient4=first2/second;
8   Console.WriteLine(quotient1);
9   Console.WriteLine(quotient2);
10  Console.WriteLine(quotient3);
11  Console.WriteLine(quotient4);
```

Output

```
1.4
1.4
1.4
1.4
```

```
1   decimal number=7.0m;// without the m it thinks its a double value!! m is for decimal
2   Console.WriteLine(number);
```

Mathematical operations - Mod/Div

```
1   Console.WriteLine($"Modulus of 200 / 5 : {200 % 5}");
2   Console.WriteLine($"Modulus of 7 / 5 : {7 % 5}");
3
4   Console.WriteLine("Modulus of 200 / 5 :"+ (200 % 5));
5   Console.WriteLine("Modulus of 7 / 5 :"+ (7 % 5));
```

```
int value1 = 3 + 4 * 5;
int value2 = (3 + 4) * 5;
Console.WriteLine(value1);
Console.WriteLine(value2);
```

```
23
35
```

Increment and Decrement

```
1   int value = 1;
2
3   value = value + 1;
4   Console.WriteLine("First increment: " + value);
5
6   value += 1;
7   Console.WriteLine("Second increment: " + value);
8
9   value++;
10  Console.WriteLine("Third increment: " + value);
11
12  value = value - 1;
13  Console.WriteLine("First decrement: " + value);
14
15  value -= 1;
16  Console.WriteLine("Second decrement: " + value);
17
18  value--;
19  Console.WriteLine("Third decrement: " + value);
```

Calling methods and instances - Intro to Classes and Objects

```
// See https://aka.ms/new-console-template for more information
Random dice = new ();
int roll1 = Random.Next(1, 7);
Console.WriteLine(roll1);
int roll2 = dice.Next(1, 7);
Console.WriteLine(roll2);
int roll3 = dice.Next(1, 7);
Console.WriteLine(roll3);
int roll4 = dice.Next(1, 7);
Console.WriteLine(roll4);
```

```
The build failed. Fix the build errors and r
PS C:\Users\hreza87\Documents\csharpprojects\
1
4
2
6
PS C:\Users\hreza87\Documents\csharpprojects\
C:\Users\hreza87\Documents\csharpprojects\Cs
non-static field, method, or property 'Rando
ct.csproj]
```

Some methods require an instance of a class, i.e object and cant be accessing classes directly such as next method.

Some methods are preconfigured to accept different parameters to avoid errors. As you can see below this concept is called over loaded.

```
Random dice = new Random();
int roll1 = dice.Next();
int roll2 = dice.Next(101);
int roll3 = dice.Next(50, 101);

Console.WriteLine($"First roll: {roll1}");
Console.WriteLine($"Second roll: {roll2}");
Console.WriteLine($"Third roll: {roll3}");
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\hreza87\Documents\csharpprojects\CsharpPr
First roll: 702623704
Second roll: 61
Third roll: 51
```

```
int firstValue = 500;
int secondValue = 600;
int largerValue = Math.Max(firstValue,secondValue);
Console.WriteLine(largerValue);
```

(local variable) int secondValue

https://learn.microsoft.com/en-us/dotnet/api/system.random.next?

Accessing Dotnet Library to find out more about functions and parameters etc

….

Selection in C# , If and else logic statements.

```
if ((roll1 == roll2) && (roll2 == roll3))
{
    Console.WriteLine("You rolled triples! +6 bonus to total!");
    total += 6;
}
```

```
int roll1 = dice.Next(1, 7);
int roll2 = dice.Next(1, 7);
int roll3 = dice.Next(1, 7);
int total = roll1 + roll2 + roll3;
// the double line is a Boolean operator Or below
if ((roll1 == roll2) || (roll2 == roll3) || (roll1 == roll3))
{
    Console.WriteLine("You rolled doubles! +2 bonus to total!");
    total += 2;
}
// this is a boolean operator AND below
if ((roll1 == roll2) && (roll2 == roll3))
{
    Console.WriteLine("You rolled triples! +6 bonus to total!");
    total += 6;
}
```

```
84  if (total >= 15)
85  {
86      Console.WriteLine($"Dice roll: {roll1} + {roll2} + {roll3} = {total}");
87      Console.WriteLine("You win!");
88  }
89
90  else
91  {
92      Console.WriteLine($"Dice roll: {roll1} + {roll2} + {roll3} = {total}");
93      Console.WriteLine("Sorry, you lose.");
94  }
95
```

Selection If and else and Nested If statements

```
Random random = new Random();
int daysUntilExpiration = random.Next(12);
int discountPercentage = 0;

if (daysUntilExpiration==0)
    {
    Console.WriteLine("Your subscription has expired already !");
    }
else
    if (daysUntilExpiration<2)
    {
    Console.WriteLine("Your subscription expires within a day!");
    Console.WriteLine($"Renew now and save {discountPercentage+20}%!");
    }
    else
        if (daysUntilExpiration<=5)
            {
            Console.WriteLine($"Your subscription expires in {daysUntilExpiration} days");
            Console.WriteLine($"Renew now and save {discountPercentage+10}%!");
            }
        else
            if (daysUntilExpiration<=10)
                {
                Console.WriteLine("Your subscription will expire soon. Renew now!");
                }
```

```
EMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

:\Users\hreza87\Documents\csharpprojects\CsharpProjects\TestProject> dotnet run
 subscription will expire soon. Renew now!
:\Users\hreza87\Documents\csharpprojects\CsharpProjects\TestProject> dotnet run
 subscription has expired already !
:\Users\hreza87\Documents\csharpprojects\CsharpProjects\TestProject> dotnet run
 subscription expires in 5 days
w now and save 10%!
:\Users\hreza87\Documents\csharpprojects\CsharpProjects\TestProject>
```

```
Random random = new Random();
int daysUntilExpiration = random.Next(12);
int discountPercentage = 0;

if (daysUntilExpiration==0)
    {
    Console.WriteLine("Your subscription expires within a day!");
    Console.WriteLine("Renew now and save 20%!");
    }
else if (daysUntilExpiration<2)
    {
    Console.WriteLine("Your subscription expires within a day!");
    Console.WriteLine("Renew now and save 20%!");
    }
else if (daysUntilExpiration<=5)
    {
    Console.WriteLine($"Your subscription expires in {daysUntilExpiration} days");
    Console.WriteLine("Renew now and save 10%");
    }
else if (daysUntilExpiration <= 10)
    {
    Console.WriteLine("Your subscription will expire soon. Renew now!");
    }
```

For loops and iterating through an array

```
int[] numbers={1,6,90};
Console.WriteLine(numbers[0]);

string[] names = { "Rowena", "Robin", "Bao" };
foreach (var name in names) // could have used a string instead ofg a var to implicitly make sure it is a string .
{
    Console.WriteLine(name);
}


int[] inventory = { 200, 450, 700, 175, 250 };
int sum=0;
foreach (int number in inventory)
    {
    Console.WriteLine(number);
    //sum=sum+ number;
    sum+=number;
    }
Console.WriteLine(sum);

Console.WriteLine($"The sum of money in inventory is {sum} roughly!");
```

```
int[] inventory = { 200, 450, 700, 175, 250 };
int sum = 0;
int bin=0;

foreach (int items in inventory)
{
    sum += items;
    bin++;
    Console.WriteLine($"Bin {bin} = {items} items (Running total: {sum})");
}
Console.WriteLine($"We have {sum} items in inventory.");



string[] orders={"B123","C234","A345","C15","B177","G3003","C235","B179"};


foreach (string order in orders)
    {
        //Console.WriteLine(order);
        if (order.StartsWith("B"))
            {
                Console.WriteLine($"The order {order} starts with 'B'!");
            }
    }
```

Creating the student grading Application

```
int examAssignments = 5;


int[] sophiaScores = { 90, 86, 87, 98, 100, 94, 90 };
int[] andrewScores = { 92, 89, 81, 96, 90, 89 };
int[] emmaScores = { 90, 85, 87, 98, 68, 89, 89, 89 };
int[] loganScores = { 90, 95, 87, 88, 96, 96 };

string[] studentNames = {"Sophia", "Andrew", "Emma", "Logan"};


int[] studentScores = new int[10];

string currentStudentLetterGrade = "";

Console.Clear();

Console.WriteLine("Student\t\tExam Score\tOverall Grade\tExtra Credit\n");
```

```
foreach(string name in studentNames)
    {
    //Console.WriteLine($"{name}");

    string currentStudent = name;

    if(currentStudent=="Sophia")
    {
        studentScores=sophiaScores;
    }
    else if (currentStudent=="Andrew")
    {
        studentScores=andrewScores;
    }
    else if(currentStudent=="Emma")
    {
        studentScores=emmaScores;
    }
    else if(currentStudent=="Logan")
    {
        studentScores=loganScores;
    }

    int gradedAssignments = 0;    // to keep track of assignments
    int gradedExtraCreditAssignments = 0;

    int sumExamScores = 0;
    int sumExtraCreditScores = 0;

    decimal currentStudentGrade = 0;
    decimal currentStudentExamScore = 0;
    decimal currentStudentExtraCreditScore = 0;
```

```
foreach(string name in studentNames)
    foreach (int score in studentScores)
    {
        gradedAssignments = gradedAssignments + 1;
        if (gradedAssignments<=examAssignments)
            {
            sumExamScores= sumExamScores + score;

            }
        else
        {
            gradedExtraCreditAssignments=gradedExtraCreditAssignments+1;
            sumExtraCreditScores=sumExtraCreditScores+score;
        }
    }

    currentStudentExamScore = (decimal)(sumExamScores) / examAssignments;
    currentStudentExtraCreditScore = (decimal)(sumExtraCreditScores) / gradedExtraCreditAssignments;

    currentStudentGrade = (decimal)((decimal)sumExamScores + ((decimal)sumExtraCreditScores / 10)) / examAssignments;

    if (currentStudentGrade >= 97)
            currentStudentLetterGrade = "A+";

        else if (currentStudentGrade >= 93)
            currentStudentLetterGrade = "A";

        else if (currentStudentGrade >= 90)
            currentStudentLetterGrade = "A-";

        else if (currentStudentGrade >= 87)
            currentStudentLetterGrade = "B+";

        else if (currentStudentGrade >= 83)
            currentStudentLetterGrade = "B";

        else if (currentStudentGrade >= 80)
            currentStudentLetterGrade = "B-";

        else if (currentStudentGrade >= 77)
            currentStudentLetterGrade = "C+";

        else if (currentStudentGrade >= 73)
            currentStudentLetterGrade = "C";

        else if (currentStudentGrade >= 70)
            currentStudentLetterGrade = "C-";
```

```
    Console.WriteLine($"{currentStudent}\t\t{currentStudentExamScore}\t\t{currentStudentGrade}\t{currentStudentLetterGrade}\t{currentStudentExtraCreditScore} ({{{{(decimal)sumExtraCreditScores / 10) / examAssignments)} pts)");


    }
Console.WriteLine("Press the Enter key to continue");
Console.ReadLine();
```

Conditional operators- Logical negation

```
string pangram = "The quick brown fox jumps over the lazy dog.";
Console.WriteLine(pangram.Contains("fox"));
Console.WriteLine(pangram.Contains("cow"));


Console.WriteLine(pangram.Contains("fox2") == false);
Console.WriteLine(!pangram.Contains("fox"));// logical negation reverses the condition. .. a bit like a NOT operator. in and or .
```

Conditional operators- Ternary conditional operator

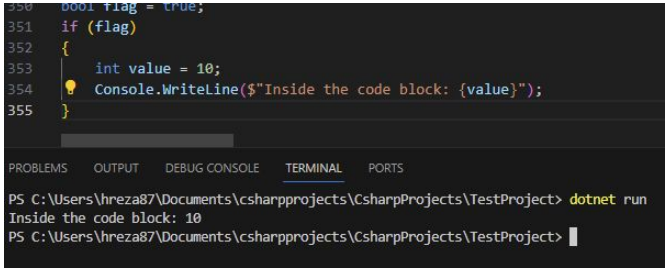
```
//METHOD 1
Random coin = new Random();
int flip = coin.Next(0, 2);
Console.WriteLine((flip == 0) ? "heads" : "tails");

//METHOD 2
string permission="Manager";
int level=15;
bool result=false;
Console.WriteLine(permission.Contains("Manager")||permission.Contains("Admin")?"you are either one but that does not matter":"YOU ARE restricited sorry ");
```

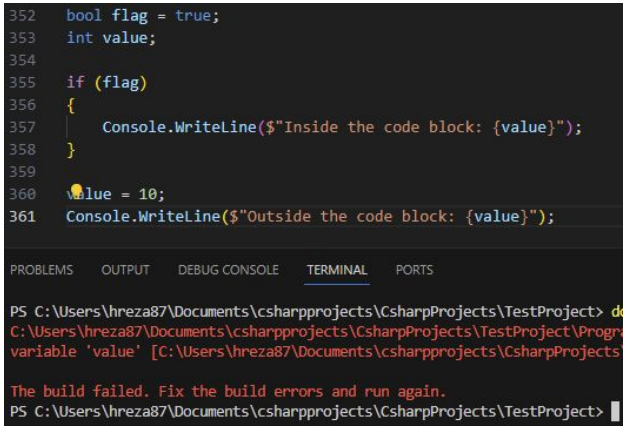
Boolean /comparison operators being used to write a program.

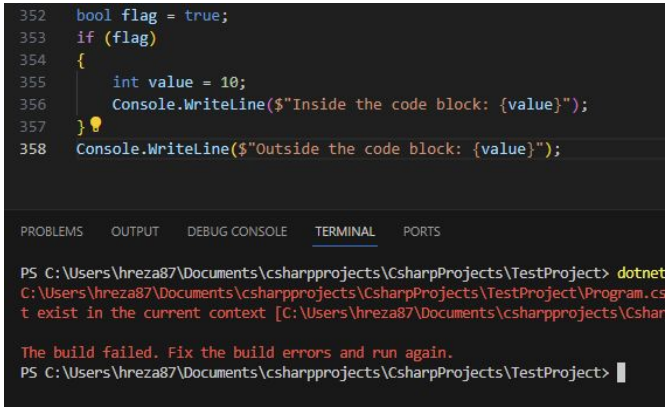
```c#
string permission = "Admin|Manager";
int level = 53;

if (permission.Contains("Admin"))
{
    if (level > 55)
    {
        Console.WriteLine("Welcome, Super Admin user.");
    }
    else
    {
        Console.WriteLine("Welcome, Admin user.");
    }
}
else if (permission.Contains("Manager"))
{
    if (level >= 20)
    {
        Console.WriteLine("Contact an Admin for access.");
    }
    else
    {
        Console.WriteLine("You do not have sufficient privileges.");
    }
}
else
{
    Console.WriteLine("You do not have sufficient privileges.");
}
```

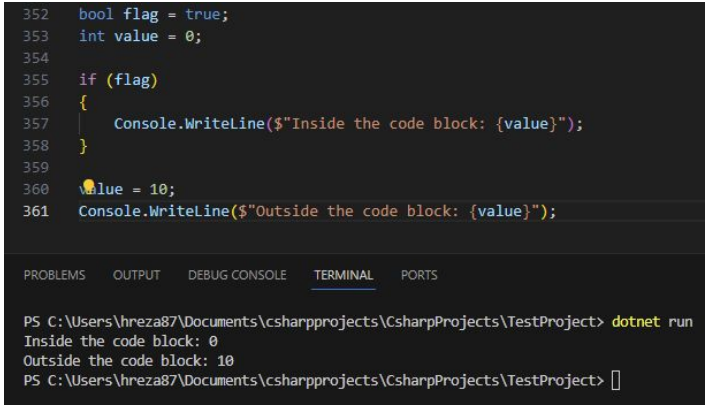
Variable Scope and examples

The variable declaration must be outside and top of code block to make it work.

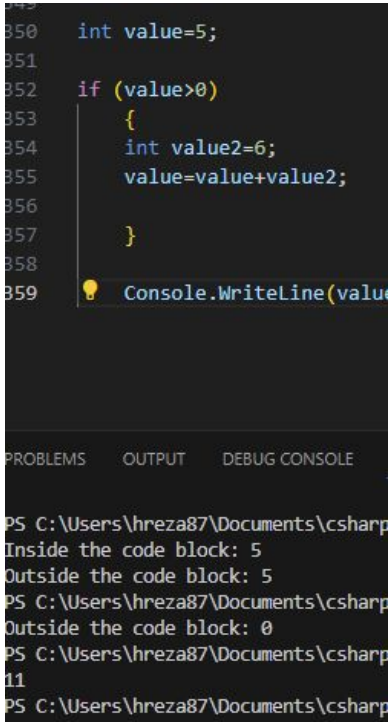




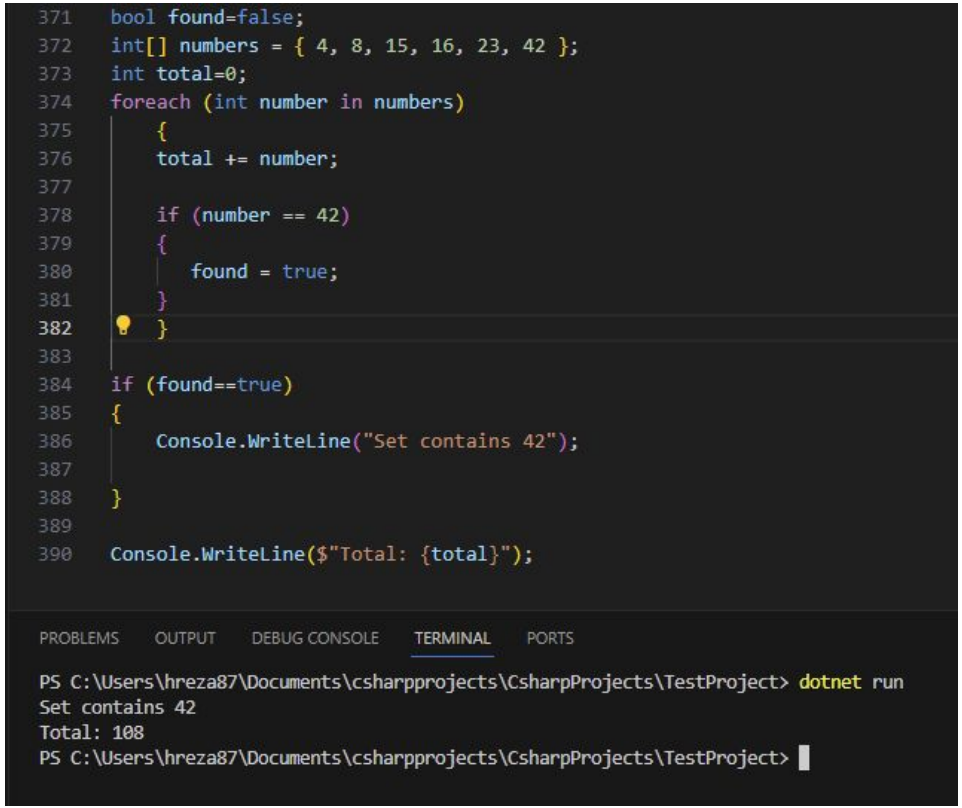




Coding blocks- assessment

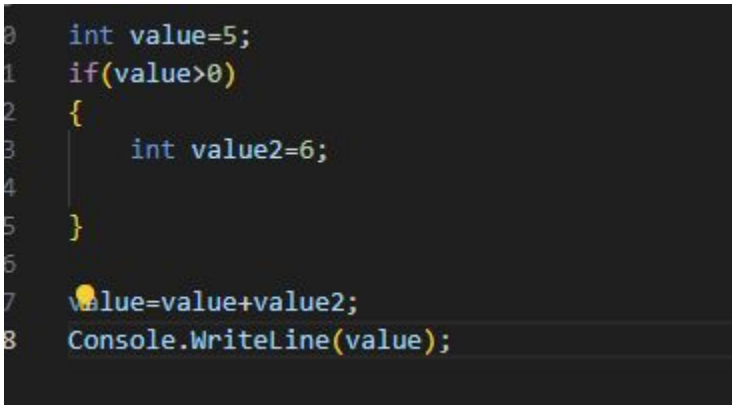


The variables need to be defined outside and assigned a default value in order for them to work outside code block when console needs to write them.



Code Blocks-Common errors

This will cause an error as value2 accessing value wont be able to, as this value was assigned in code block, in order to make it accessible by whole programs, assignment must be done outside of the code blocks.



```
int value=5;
if(value>0)
{
    int value2=6;

}

value=value+value2;
Console.WriteLine(value);
```

Selection- Switch Cases

You can use multiple cases to define certain outcome as shown in the example on the right.

```
369  string fruit="apple";
370  switch (fruit)
371  {
372      case "apple":
373          Console.WriteLine($"App will display information for apple.");
374          break;
375
376      case "banana":
377          Console.WriteLine($"App will display information for banana.");
378          break;
379
380      case "cherry":
381          Console.WriteLine($"App will display information for cherry.");
382          break;
383  }
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\TestProject> dotnet run
App will display information for apple.
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\TestProject>
```

```
369  int employeeLevel = 100;
370  string employeeName = "John Smith";
371
372  string title = "";
373
374  switch (employeeLevel)
375  {
376      case 100:
377      case 200:
378          title = "Senior Associate";
379          break;
380      case 300:
381          title = "Manager";
382          break;
383      case 400:
384          title = "Senior Manager";
385          break;
386      default:
387          title = "Associate";
388          break;
389  }
390
391  Console.WriteLine($"{employeeName}, {title}");
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\TestProject> dotnet run
App will display information for apple.
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\TestProject> dotnet run
John Smith, Senior Associate
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\TestProject> dotnet run
John Smith, Associate
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\TestProject> dotnet run
John Smith, Associate
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\TestProject> dotnet run
John Smith, Senior Associate
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\TestProject>
```

Switch Challenge

```
string sku = "01-MN-L";
string[] product = sku.Split('-');
string type = "";
string color = "";
string size = "";

switch(product[0])
{
    case "01":
        type = "Sweat shirt";
        break;
    case "02":
        type = "T-Shirt";
        break;
    case "03":
        type= "Sweat pants";
        break;
    default:
        type="other";
        break;
}
switch(product[1])
{
    case "BL":
        color = "Black";
        break;
    case "MN":
        color = "Maroon";
        break;
    default:
        color = "White";
        break;
}
switch(product[2])
{
    case "S":
        size = "Small";
        break;
    case "M":
        size = "Medium";
        break;
    case "L":
        size = "Large";
        break;
    default:
        size = "One Size Fits All";
        break;
}
Console.WriteLine($"Product: {size} {color} {type}");
```

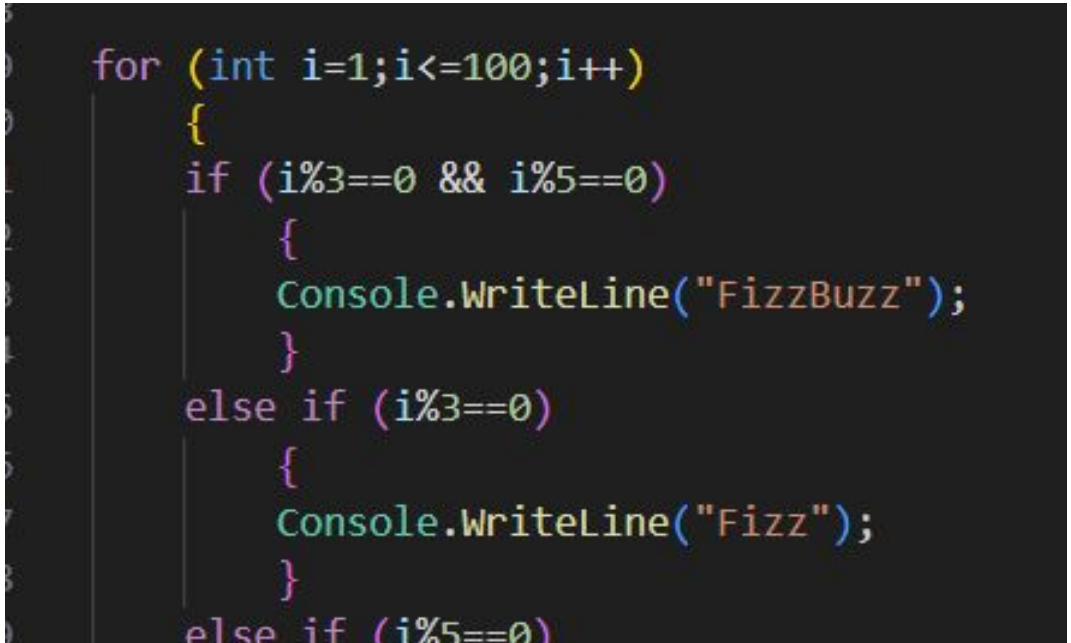
```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\TestProject> dotnet run
01
MN
L
Product: Large Maroon Sweat shirt
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\TestProject> dotnet run
Product: Large White other
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\TestProject> dotnet run
Product: Large Maroon Sweat shirt
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\TestProject>
```

```
string sku = "01-MN-L";
string type = "";
string color = "";
string size = "";
string[] product = sku.Split('-');
for (int i = 0; i < product.Length; i++)
    {
    Console.WriteLine(product[i]);
    switch(i)
        {
        case(0):
            switch(product[i])
            {
            case "01":
                type = "Sweat shirt";
                break;
            case "02":
                type = "T-Shirt";
                break;
            case "03":
                type= "Sweat pants";
                break;
            default:
                type="other";
                break;
            }
            break;
        case(1):
            switch(product[i])
            {
            case "BL":
                color = "Black";
                break;
            case "MN":
                color = "Maroon";
                break;
            default:
                color = "White";
                break;
            }
            break;
        case(2):
            switch(product[i])
            {
            case "S":
                size = "Small";
                break;
            case "M":
                size = "Medium";
                break;
            case "L":
                size = "Large";
                break;
            default:
                size = "One Size Fits All";
                break;
            }
            break;
        }
    }
Console.WriteLine($"Product: {size} {color} {type}");
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

pProjects\TestProject\TestProject.csproj]
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\TestProject> dotnet run
01
MN
L
Product: Large Maroon Sweat shirt
```

Iteration For, ForEach,

From what I have gathered for each item in list is good for accessing however for modifying a a list , the best thing is using a for.



Do while and While

```
680   Random random = new Random();
681   int current = 0;
682
683   do
684   {
685       current = random.Next(1, 11);
686       Console.WriteLine(current);
687   } while (current != 7);
```

PROBLEMS OUTPUT TERMINAL ··· powershell

```
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\Te
t> dotnet run
9
6
9
4
2
9
8
2
9
6
1
3
6
3
9
6
2
10
2
8
7
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\Te
t>
```

```
*/
Random random = new Random();
int current=4;
while (current >= 3)
{
    Console.WriteLine(current);
    current = random.Next(1, 11);
}
Console.WriteLine($"Last number: {current}");
```

Using a while loop

Tomake a game such as monster vs human.

While can use multiple comparisons .

Thesame could be done using do while

```
Random random=new Random();
int hero=10;
int monster=10;
bool heroturn=true;
while ((hero>0)&&(monster>0))
{
    int attackpoints = random.Next(1, 10);
    if (heroturn==true)
        {
        monster=monster-attackpoints;
        Console.WriteLine($"Monster was damaged and lost {attackpoints} health and now has {monster}health");
        heroturn=false;
        }
    else
        {
        hero=hero-attackpoints;
        Console.WriteLine($"Hero was damaged and lost {attackpoints} health and now has {hero}health");
        heroturn=true;
        }
}
if (hero>monster)
{
    Console.WriteLine("Hero Wins");
}
else
{
    Console.WriteLine("Monster wins");
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Hero was damaged and lost 9 health and now has -5health
Monster wins
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\TestProject> dotnet run
Monster was damaged and lost 3 health and now has 7health
Hero was damaged and lost 8 health and now has 2health
Monster was damaged and lost 4 health and now has 3health
Hero was damaged and lost 1 health and now has 1health
Monster was damaged and lost 1 health and now has 2health
Hero was damaged and lost 9 health and now has -8health
Monster wins
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\TestProject>
```

Do While vs While loop

```
745  string word="";
746  bool validcheck=false;
747  Console.WriteLine("Enter a string containing min of 3 letters");
748
749  do
750  {
751      word=Console.ReadLine();
752      if (word != null)
753          {
754          if (word.Length>=3)
755              {
756              validcheck=true;
757              }
758
759          else
760              {
761              Console.WriteLine("invalid entry, try again:");
762              validcheck=false;
763              }
764          }
765  }while (validcheck==false);
766
767  Console.WriteLine("Well done you have entered the 3 or more ");
768
```

```
string? word;
bool validcheck=false;
Console.WriteLine("enter a string that is more then 3 letters :");
while (validcheck==false)
{
    word=Console.ReadLine();
    if (word !=null)
        {
        if (word.Length >= 3)
            {
            validcheck=true;
            }

        else Console.WriteLine("Invalid entry, please try again :");
        }
    else
        {
        Console.WriteLine("Invalid entry, please try again :");
        validcheck=false;

        }
}

Console.WriteLine("You have entered a valid word");
```

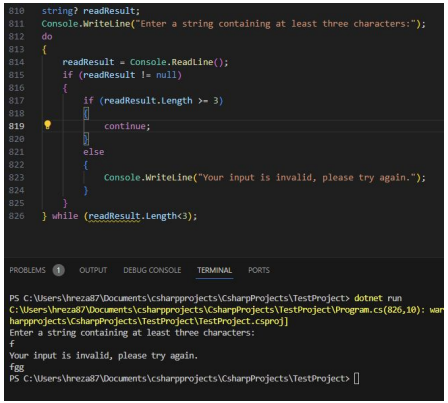
Do While and While

```
801    string? readResult;
802    Console.WriteLine("Enter a string:");
803    do
804    {
805        readResult = Console.ReadLine();
806    } while (readResult == null);
```

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORT

PS C:\Users\hreza87\Documents\csharpprojects\Cshar

Usage: dotnet [options]
Usage: dotnet [path-to-application]

Options:
  -h|--help         Display help.
  --info            Display .NET information.
  --list-sdks       Display the installed SDKs.
  --list-runtimes   Display the installed runtimes

path-to-application:
  The path to an application .dll file to execute.
PS C:\Users\hreza87\Documents\csharpprojects\Cshar
Enter a string:
haider rizvi
```

```
810  string? readResult;
811  bool validEntry = false;
812  Console.WriteLine("Enter a string containing at least three characters:");
813  do
814  {
815      readResult = Console.ReadLine();
816      if (readResult != null)
817      {
818          if (readResult.Length >= 3)
819          {
820              validEntry = true;
821          }
822          else
823          {
824              Console.WriteLine("Your input is invalid, please try again.");
825          }
826      }
827  } while (validEntry == false);
```

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\TestProject> dotnet run
Enter a string containing at least three characters:
43
Your input is invalid, please try again.
4
Your input is invalid, please try again.
5
Your input is invalid, please try again.
4546
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\TestProject>
```



Do While (Validation checks) Code Project 1

```
845  string? readResult;
846  int numericValue=0;
847  bool validNumber = false;
848  bool fullvalidity=false;
849  Console.WriteLine("Enter a valid integer between 5 and 10");
850  do
851  {
852      readResult=Console.ReadLine();
853      validNumber = int.TryParse(readResult, out numericValue);
854      if (validNumber)
855          {
856              if (numericValue>5 && numericValue<10)
857              {
858                  fullvalidity=true;
859              }
860              else
861              {
862                  Console.WriteLine($"You entered {numericValue}, please try between 5 and 10!");
863              }
864          }
865      else
866          {
867              Console.WriteLine("Sorry, you entered an invalid number, please try again!");
868          }
869
870
871  } while (fullvalidity==false);
872
873  Console.WriteLine($"Your input of  {numericValue} has been accepted !");
```

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\TestProject> dotnet run
Enter a valid integer between 5 and 10
hello world
Sorry, you entered an invalid number, please try again!
4
You entered 4, please try between 5 and 10!
5
You entered 5, please try between 5 and 10!
6
Your input of  6 has been accepted !
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\TestProject>
```

While Loop for validation check -Code Project 1

```
880  string? readResult;
881  int numericValue=0;
882  bool validNumber = false;
883  bool fullvalidity=false;
884
885  while (fullvalidity ==false)
886  {
887      Console.WriteLine("Enter a valid integer between 5 and 10");
888      readResult=Console.ReadLine();
889      validNumber = int.TryParse(readResult, out numericValue);
890      if (validNumber)
891          {
892              if (numericValue>5 && numericValue<10)
893              {
894                  fullvalidity=true;
895              }
896
897              else
898              {
899                  Console.WriteLine("Enter a number in the correct range pl
900              }
901          }
902      else
903          {
904              Console.WriteLine("enter a valid number cheers.");
905          }
906
907
908  }
909
910  Console.WriteLine($"You have entered a valid integer i.e {numericValue}");
911
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\TestProject> dotnet run
Enter a valid integer between 5 and 10
haider
enter a valid number cheers.
Enter a valid integer between 5 and 10
4
Enter a number in the correct range please 5-10!
Enter a valid integer between 5 and 10
6
You have entered a valid integer i.e 6
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\TestProject>
```

Code Project 2

Code Project 3

Basic version going through one string.

```
1045  string myString = "I like pizza. I like roast chicken. I like salad";
1046  int position = 0;
1047
1048  while (true)
1049  {
1050      position = myString.IndexOf(".");
1051
1052      if (position != -1)
1053      {
1054          string substring = myString.Substring(0, position); // Extract substring and remove leading/trailing spaces
1055          Console.WriteLine(substring);
1056          myString = myString.Substring(position + 1); // Update myString to start after the period
1057      }
1058      else
1059      {
1060          Console.WriteLine(myString); // Print the remaining part of myString (after the last period)
1061          break;
1062      }
1063  }
```

Code Project 3 - Main Challenge

```
975  string[] myStrings = {"I like pizza. I like roast chicken. I like salad", "I like all three of the menu choices"};
976  for (int i = 0; i < myStrings.Length; i++)
977  {
978      string myString=myStrings[i];
979
980      int position = 0;
981
982      while (true)
983      {
984          position = myString.IndexOf(".");
985
986          if (position != -1)
987          {
988              string substring = myString.Substring(0, position); // Extract substring and remove leading/trailing spaces
989              Console.WriteLine(substring.TrimStart());
990              myString = myString.Substring(position + 1).TrimStart(); // Update myString to start after the period
991          }
992          else
993          {
994              Console.WriteLine(myString.TrimStart()); // Print the remaining part of myString (after the last period)
995              break;
996          }
997      }
998  }
999  /*
1000 string myString = "I like pizza. I like roast chicken. I like salad";
1001 int position=0;
```

Two Dimensional Arrays in More Detail

```
string[][] jaggedArray = new string[][]
{
    new string[] { "one1", "two1", "three1", "four1", "five1", "six1" },
    new string[] { "one2", "two2", "three2", "four2", "five2", "six2" },
    new string[] { "one3", "two3", "three3", "four3", "five3", "six3" },
    new string[] { "one4", "two4", "three4", "four4", "five4", "six4" },
    new string[] { "one5", "two5", "three5", "four5", "five5", "six5" },
    new string[] { "one6", "two6", "three6", "four6", "five6", "six6" },
    new string[] { "one7", "two7", "three7", "four7", "five7", "six7" },
    new string[] { "one8", "two8", "three8", "four8", "five8", "six8" }
};

foreach (string[] array in jaggedArray)
{
    foreach (string value in array)
    {
        Console.WriteLine(value);
    }
    Console.WriteLine();
}
```

Needs populating with notes properly.

Data Types - Value based and Reference Based.

There are two integral data types
Signed and unsigned.

```
CsharpProjects > TestProject > Program.cs
1   Console.WriteLine("Signed integral types:");
2
3   Console.WriteLine($"sbyte  : {sbyte.MinValue} to {sbyte.MaxValue}");
4   Console.WriteLine($"short  : {short.MinValue} to {short.MaxValue}");
5   Console.WriteLine($"int    : {int.MinValue} to {int.MaxValue}");
6   Console.WriteLine($"long   : {long.MinValue} to {long.MaxValue}");
7
8   Console.WriteLine("");
9   Console.WriteLine("Unsigned integral types:");
10
11  Console.WriteLine($"byte   : {byte.MinValue} to {byte.MaxValue}");
12  Console.WriteLine($"ushort : {ushort.MinValue} to {ushort.MaxValue}");
13  Console.WriteLine($"uint   : {uint.MinValue} to {uint.MaxValue}");
14  Console.WriteLine($"ulong  : {ulong.MinValue} to {ulong.MaxValue}");
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

int    : -2147483648 to 2147483647
long   : -9223372036854775808 to 9223372036854775807
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\Module4\CsharpProjects\TestProject> dotnet run
Signed integral types:
sbyte  : -128 to 127
short  : -32768 to 32767
int    : -2147483648 to 2147483647
long   : -9223372036854775808 to 9223372036854775807

Unsigned integral types:
byte   : 0 to 255
ushort : 0 to 65535
uint   : 0 to 4294967295
ulong  : 0 to 18446744073709551615
```

Floating point numbers , Decimals,

Floats and doubles good for approximate figures, where as for precisions such as financial transactions, space , decimals give 28 to 29 digit precision.

```
16  Console.WriteLine("");
17  Console.WriteLine("Floating point types:");
18  Console.WriteLine($"float  : {float.MinValue} to {float.MaxValue} (with ~6-9 digits of precision)");
19  Console.WriteLine($"double : {double.MinValue} to {double.MaxValue} (with ~15-17 digits of precision)");
20  Console.WriteLine($"decimal: {decimal.MinValue} to {decimal.MaxValue} (with 28-29 digits of precision)");
```

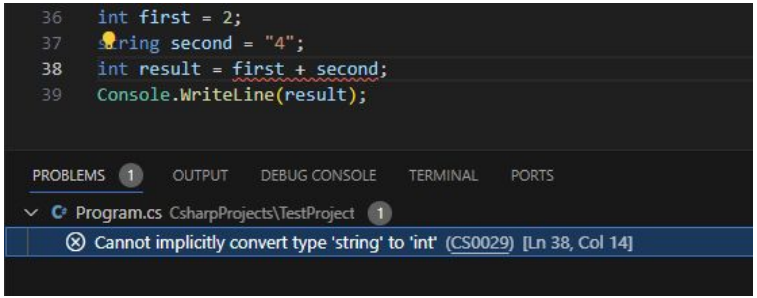
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

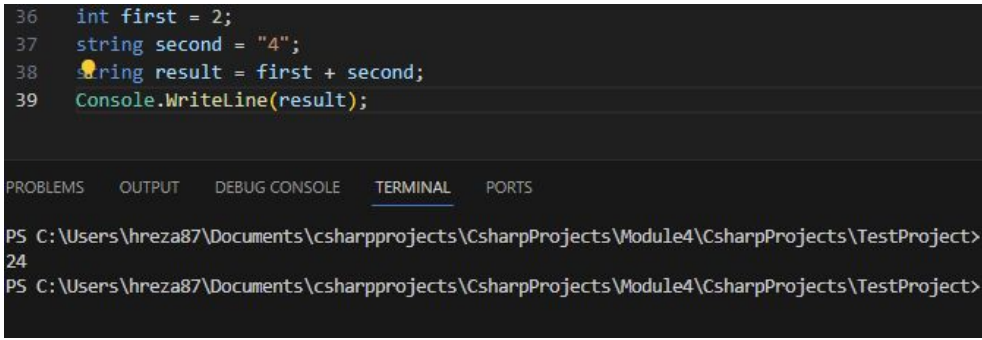
```
int    : -2147483648 to 2147483647
long   : -9223372036854775808 to 9223372036854775807
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\Module4\CsharpProjects\TestProject> dotnet run
Signed integral types:
sbyte  : -128 to 127
short  : -32768 to 32767
int    : -2147483648 to 2147483647
long   : -9223372036854775808 to 9223372036854775807

Unsigned integral types:
byte   : 0 to 255
ushort : 0 to 65535
uint   : 0 to 4294967295
ulong  : 0 to 18446744073709551615
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\Module4\CsharpProjects\TestProject> dotnet run
Signed integral types:
sbyte  : -128 to 127
short  : -32768 to 32767
int    : -2147483648 to 2147483647
long   : -9223372036854775808 to 9223372036854775807

Unsigned integral types:
byte   : 0 to 255
ushort : 0 to 65535
uint   : 0 to 4294967295
ulong  : 0 to 18446744073709551615

Floating point types:
float  : -3.4028235E+38 to 3.4028235E+38 (with ~6-9 digits of precision)
double : -1.7976931348623157E+308 to 1.7976931348623157E+308 (with ~15-17 digits of precision)
decimal: -79228162514264337593543950335 to 79228162514264337593543950335 (with 28-29 digits of precision)
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\Module4\CsharpProjects\TestProject>
```

Casting /Converting Data types for relevant applications

Here are some issues encountered:

1. Adding integer and string , storing result as integer causes issues.
2. Adding integer and string, storing result as string makes compiler perform the task. You can see concatenation has happened.





Widening and Narrowing Conversion of datatypes/variables

Widening conversion is when a data type that holds smaller value is being converted to the one that holds larger value.

Narrowing would be the opposite I suppose.

You need to ensure you use a (casting) operator a s shown.

```
35  int myInt = 3;
36  Console.WriteLine($"int: {myInt}");
37
38  decimal myDecimal = myInt;
39  Console.WriteLine($"decimal: {myDecimal}");
40
41
```

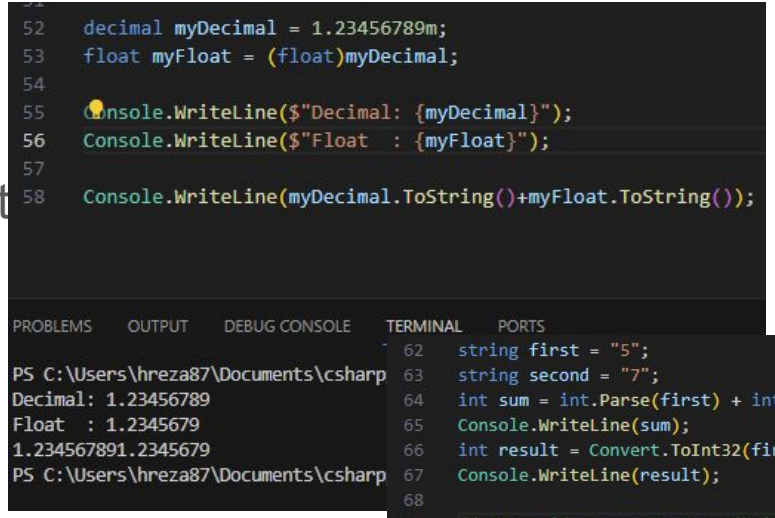
```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\hreza87\Documents\csharpprojects\CsharpProj
int: 3
decimal: 3
```

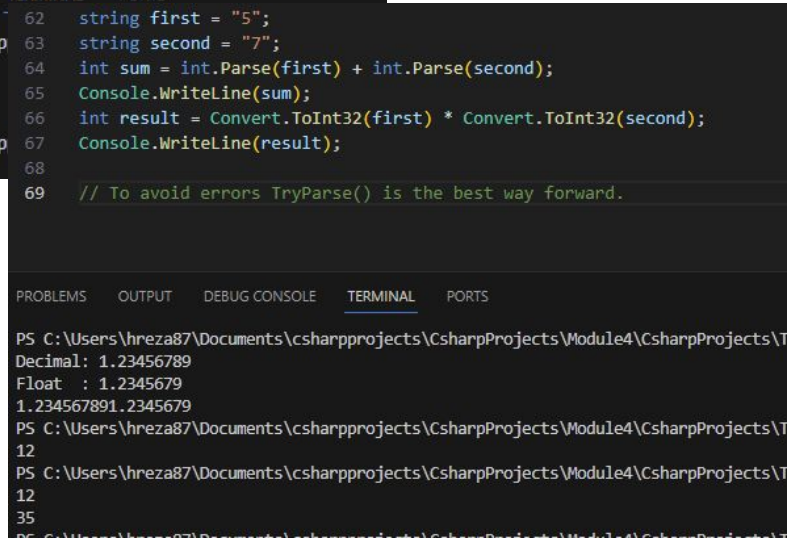
```
52  decimal myDecimal = 1.23456789m;
53  float myFloat = (float)myDecimal;
54
55  Console.WriteLine($"Decimal: {myDecimal}");
56  Console.WriteLine($"Float  : {myFloat}");
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\Module
decimal: 3.14
int: 3
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\Module
Decimal: 1.23456789
Float  : 1.2345679
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\Module
```

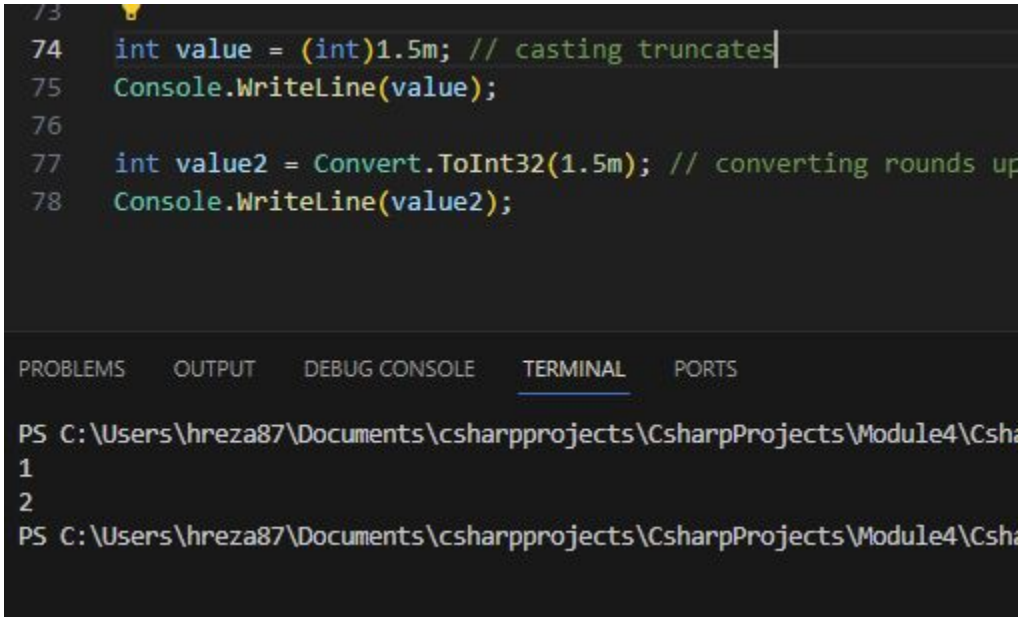
Data type conversions -ToString() , Parse()....helper methods.

1. variable.ToString()
2. int.Parse(variable) ..convert numbers into integers.
3. Convert.ToInt32(Variable)





Casting and Converting a decimal to Int -Comparison



Try Parse in more depth.

int.TryParse(value, out result) working out fine will generate a value of True or False.

```
string value = "102rer";
The boolean condition would generate:False
```

```
84  //string name = "Bob";
85  //Console.WriteLine(int.Parse(name)); This will raise the error due to variable not in the correct format.
86  string value = "102";
87  int result = 0;
88  if (int.TryParse(value, out result))
89  {
90      Console.WriteLine($"Measurement: {result}");
91  }
92  else
93  {
94      Console.WriteLine("Unable to report the measurement.");
95  }
96
97
98  Console.WriteLine($"The boolean condition would generate:{int.TryParse(value, out result)}");
99
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\Module4\CsharpProjects\TestProject> dotnet run
Measurement: 102
The boolean condition would generate:True
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\Module4\CsharpProjects\TestProject>
```

Error Handling when converting data types

```
//string name = "Bob";
//Console.WriteLine(int.Parse(name)); This will raise the error due to variable not in the correct format.
string value = "20";
int result = 0;
if (int.TryParse(value, out result))
{
    Console.WriteLine($"Measurement: {result}");
}
else
{
    Console.WriteLine("Unable to report the measurement.");
}

if (result > 0)
    Console.WriteLine($"Measurement (w/ offset): {50 + result}");
```

Combining String and integers validation

Microsoft way vs my way on the right

```
string[] values = { "12.3", "45", "ABC", "11", "DEF" };

decimal total = 0m;
string message = "";

foreach (var value in values)
{
    decimal number; // stores the TryParse "out" value
    if (decimal.TryParse(value, out number))
    {
        total += number;
    } else
    {
        message += value;
    }
}

Console.WriteLine($"Message: {message}");
Console.WriteLine($"Total: {total}");
```

```
105  string[] values = { "12.332", "45", "ABC", "11", "DEF" };
106  string word="";
107  decimal sum=0;
108  decimal number=0;
109
110  foreach (string value in values)
111      {
112          //Console.WriteLine(value);
113          if (decimal.TryParse(value,out number ))
114          {
115              sum=sum+number;
116
117          }
118          else
119              {
120                  word=word+value;
121              }
122      }
123
124  Console.WriteLine($"Message:{word}");
125  Console.WriteLine($"Total:{sum}");
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\Module4\Csha
Message:ABCDEF
Total:68.332
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\Module4\Csha
Message:ABCDEF
Total:68.332
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\Module4\Csha
```

Further exercise converting data type during calculations

If int is being multiple ro divided by float, it will automatically be converted, no need to cas

My apologies for the confusion. You're correct; casting `value1` to `float` is unnecessary in this context because the division operation between a `float` and an `int` will automatically promote the `int` to a `float`. Here's the corrected version:

csharp Copy code

```csharp
int value1 = 12;       // Initializing an integer variable value1 with value 12
decimal value2 = 6.2m; // Initializing a decimal variable value2 with value 6.2
float value3 = 4.3f;   // Initializing a float variable value3 with value 4.3

float result3 = value3 / value1; // Dividing value3 by value1 (value1 will be implici
Console.WriteLine($"Divide value3 by value1, display the result as a float: {result3}
```

In C#, when you divide a `float` by an `int`, the `int` will be implicitly converted to a `float` before the division operation occurs. Therefore, explicitly casting `value1` to `float` is unnecessary in this case. The code will work correctly without it.

output math operations as specific number types challenge

```
129   int value1 = 12;
130   decimal value2 = 6.2m;
131   float value3 = 4.3f;
132
133   // Your code here to set result1
134   // Hint: You need to round the result to nearest integer (don't just truncate)
135
136   int result1 =value1/Convert.ToInt32(value2);
137   Console.WriteLine($"Divide value1 by value2, display the result as an int: {result1}");
138
139   // Your code here to set result2
140
141   decimal result2=value2/ (decimal)value3;
142   Console.WriteLine($"Divide value2 by value3, display the result as a decimal: {result2}");
143   💡
144   float result3=value3/(float)value1;
145   Console.WriteLine($"Divide value3 by value1, display the result as a float: {result3}");
```

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\Module4\CsharpProjects\TestProject> dotnet run
Divide value1 by value2, display the result as an int: 2
Divide value2 by value3, display the result as a decimal: 1.4418604651162790697674418605
Divide value3 by value1, display the result as a float: 0.35833335
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\Module4\CsharpProjects\TestProject>
```

C# Arrays and operations

Clear(arrayname, starting index, clearing 2 elements from ther.e

```
168  string[] pallets = { "B14", "A11", "B12", "A13" };
169  Console.WriteLine("");
170
171  Array.Clear(pallets, 0, 2);
172  Console.WriteLine($"Clearing 2 ... count: {pallets.Length}");
173  foreach (var pallet in pallets)
174  {
175      Console.WriteLine($"-- {pallet}");
176  }
177
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\Module4\CsharpProjects\TestProject> dotnet run

Clearing 2 ... count: 4
--
--
-- B12
-- A13
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\Module4\CsharpProjects\TestProject>
```

```
149  string[] pallets = { "B14", "A11", "B12", "A13" };
150
151  Console.WriteLine("Sorted...");
152  Array.Sort(pallets);
153  foreach (var pallet in pallets)
154  {
155      Console.WriteLine($"-- {pallet}");
     }

     Console.WriteLine("");
     Console.WriteLine("Reversed...");
     Array.Reverse(pallets);
     foreach (var pallet in pallets)
     {
         Console.WriteLine($"-- {pallet}");
     }
```

```
IS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

Jsers\hreza87\Documents\csharpprojects\CsharpProjects\Module4\Csha
...

ed...

Jsers\hreza87\Documents\csharpprojects\CsharpProjects\Module4\Csha
```

Resize Array

```
205  string[] pallets = { "B14", "A11", "B12", "A13" };
206  Console.WriteLine("");
207
208  Array.Clear(pallets, 0, 2);
209  Console.WriteLine($"Clearing 2 ... count: {pallets.Length}");
210  foreach (var pallet in pallets)
211  {
212      Console.WriteLine($"-- {pallet}");
213  }
214
215  Console.WriteLine("");
216  Array.Resize(ref pallets, 6);
217  Console.WriteLine($"Resizing 6 ... count: {pallets.Length}");
218
219  pallets[4] = "C01";
220  pallets[5] = "C02";
221
222  foreach (var pallet in pallets)
223  {
224      Console.WriteLine($"-- {pallet}");
225  }
226
227  Console.WriteLine("");
228  Array.Resize(ref pallets, 3);
229  Console.WriteLine($"Resizing 3 ... count: {pallets.Length}");
230
231  foreach (var pallet in pallets)
232  {
233      Console.WriteLine($"-- {pallet}");
234  }
235
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\Module4\CsharpProjects\Test

Clearing 2 ... count: 4
--
--
-- B12
-- A13

Resizing 6 ... count: 6
--
--
-- B12
-- A13
-- C01
-- C02

Resizing 3 ... count: 3
--
--
-- B12
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\Module4\CsharpProjects\Test
```

Split() and Join()

Trimming a string

```
string originalString = "  This is a string with whitespace.   ";
string trimmedString = originalString.Trim();
```

```
239  using System.Dynamic;
240
241  string name=" Haider Rizvi";
242  Console.WriteLine(name);
243  string name2=name.Trim();// removes white spacing
244  Console.WriteLine(name2);
245  char[] letters=name2.ToCharArray();//converts a string into array of characters
246  string result=String.Join(",",letters);//creates new string using by including commas,
247
248  Console.WriteLine(result);
249
250  string[] items=result.Split(","); // creates a array of strings in a string by using comma as a delimiter.
251  foreach (string item in items)
252      {
253      Console.WriteLine(item);
254      }
```

```
PS C:\Users\hreza87\Documents\csh
 Haider Rizvi
Haider Rizvi
H,a,i,d,e,r, ,R,i,z,v,i
H
a
i
d
e
r

R
i
z
v
i
PS C:\Users\hreza87\Documents\csh
```

Split,Join Challenge

```
258  string pangram = "The quick brown fox jumps over the lazy dog";
259  string[] items=pangram.Split(" ");
260  string word="";
261  foreach (string item in items)
262      {
263          char[] letters=item.ToCharArray();
264          Array.Reverse(letters);
265          string result=new string(letters);
266          word = word+" "+ result;
267
268      }
269
270  Console.WriteLine(word);
```

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\Module4\CsharpProjects\Te
 ehT kciuq nworb xof spmuj revo eht yzal god
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\Module4\CsharpProjects\Te
```

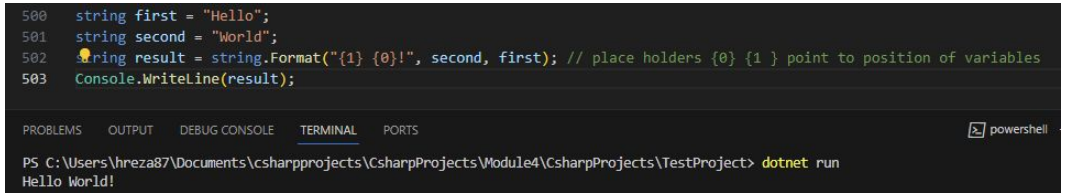
```
270  Console.WriteLine(word);
271  string pangram = "The quick brown fox jumps over the lazy dog";
272  // Step 1
273  string[] message = pangram.Split(' ');
274  //Step 2
275  string[] newMessage = new string[message.Length];
276  // Step 3
277  for (int i = 0; i < message.Length; i++)
278  {
279      char[] letters = message[i].ToCharArray();
280      Array.Reverse(letters);
281      newMessage[i] = new string(letters);
282  }
283  //Step 4
284  string result = String.Join(" ", newMessage);
285  Console.WriteLine(result);
```

Parsing strings and verifying their length

```
482  string orderStream = "B123,C234,A345,C15,B177,G3003,C235,B179";
483  string[] orders=orderStream.Split(",");
484  Array.Sort(orders);
485
486  for (int i=0; i <orders.Length;i++)
487      if (orders[i].Length==4)
488      {
489          Console.WriteLine(orders[i]);
490      }
491      else
492          Console.WriteLine($"{orders[i]} -- Error");
493
494
495
496  int[] start = {2,2,2,2};
497  int[] end = {4,4,4,4};
498
```

```
A345
B123
B177
B179
C15 -- Error
C234
C235
G3003 -- Error
```

Format Alphanumeric Data for Presentation String Composite Formatting , bottom is String Interpolation




```
510    string first = "Hello";
511    string second = "World";
512    Console.WriteLine($"{first} {second}!");
513    Console.WriteLine($"{second} {first}!");
514    Console.WriteLine($"{first} {first} {first}!");
```

Formatting Currency ,Numbers, %

```
517   decimal price = 123.45m;
518   int discount = 50;
519   Console.WriteLine($"Price: {price:C} (Save {discount:C})");
520
521   decimal measurement = 123456.78912m;
522   Console.WriteLine($"Measurement: {measurement:N} units");
523
524   decimal tax = .36785m;
525   Console.WriteLine($"Tax rate: {tax:P2}");
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\Module4\CsharpP
Price: £123.45 (Save £50.00)
Measurement: 123,456.79 units
Tax rate: 36.79%
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\Module4\CsharpP
```

Combining formatting approaches.

```
528   decimal price = 67.55m;
529   decimal salePrice = 59.99m;
530
531   string yourDiscount = String.Format("You saved {0:C2} off the regular {1:C2} price. ", (price - salePrice), price);
532
533   yourDiscount += $"A discount of {((price - salePrice)/price):P2}!"; //inserted
534   Console.WriteLine(yourDiscount);
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS                                                    powershell

PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\Module4\CsharpProjects\TestProject> dotnet run
Price: £123.45 (Save £50.00)
Measurement: 123,456.79 units
Tax rate: 36.79%
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\Module4\CsharpProjects\TestProject>
```

Recap

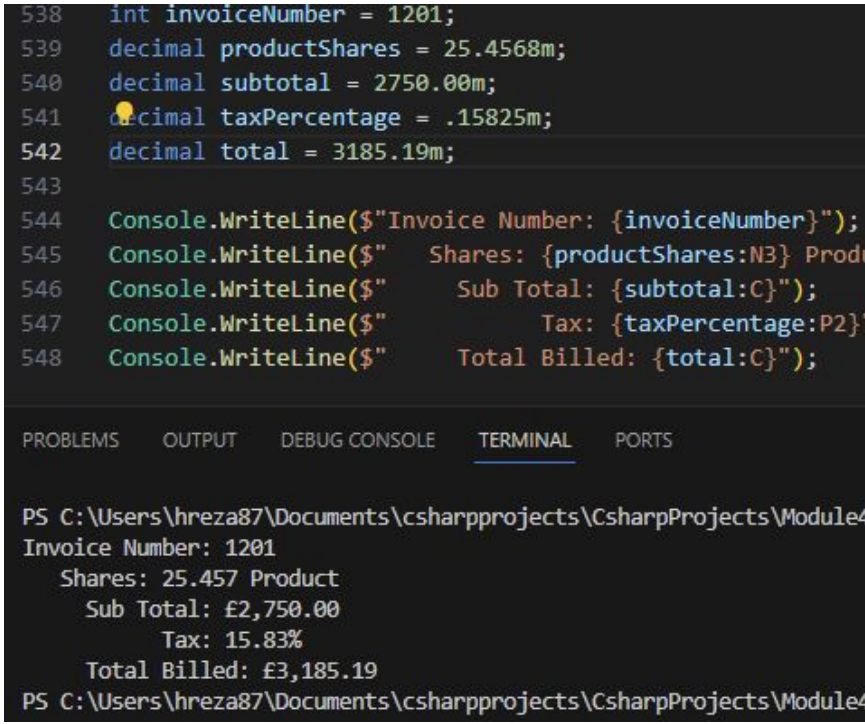
Here are most important takeaways from this unit about string formatting:

- You can use composite formatting or string interpolation to format strings.
- With **composite formatting**, you use a string template containing one or more replacement tokens in the form `{0}`. You also supply a list of arguments that are matched with the replacement tokens based on their order. Composite formatting works when using `string.Format()` or `Console.WriteLine()`.
- With **string interpolation**, you use a string template containing the variable names you want replaced surrounded by curly braces. Use the `$` directive before the string template to indicate you want the string to be interpolated.
- Format currency using a `:C` specifier.
- Format numbers using a `:N` specifier. Control the precision (number of values after the decimal point) using a number after the `:N` like `{myNumber:N3}`.
- Format percentages using the `:P` format specifier.
- Formatting currency and numbers depend on the end user's culture, a five character code that includes the user's country/region and language (per the settings on their computer).

Exploring String Interpolation

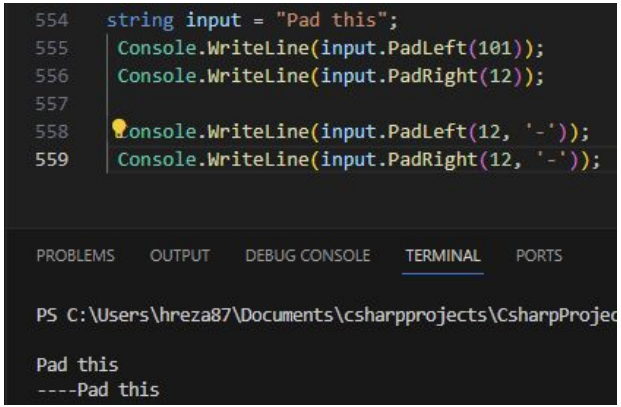
The N3 is decimal places after decimal.

P2 is how many decimal places after.



Discover Padding and Alignment

- Methods that add blank spaces for formatting purposes (`PadLeft()`, `PadRight()`)
- Methods that compare two strings or facilitate comparison (`Trim()`, `TrimStart()`, `TrimEnd()`, `GetHashcode()`, the `Length` property)
- Methods that help you determine what's inside of a string, or even retrieve just a part of the string (`Contains()`, `StartsWith()`, `EndsWith()`, `Substring()`)
- Methods that change the content of the string by replacing, inserting, or removing parts (`Replace()`, `Insert()`, `Remove()`)
- Methods that turn a string into an array of strings or characters (`Split()`, `ToCharArray()`)



Use the String's Index of() and Substring() helper methods

Index of (this is where you specify single character to find index of)

Substring(parameter 1,Parameter2) p1 is starting index, p2 is length of string from there.

```
687  string message = "Filter out stuff between (Howdy Stranger)";
688
689  int openingPosition = message.IndexOf('(');
690  int closingPosition = message.IndexOf(')');
691
692  Console.WriteLine(openingPosition);
693  Console.WriteLine(closingPosition);
694
695  int length = closingPosition - openingPosition;
696  Console.WriteLine(message.Substring(openingPosition+1, length-1));//where to start from first, how many characters there or
```

```
PS C:\Users\hreza87
25
40
Howdy Stranger
PS C:\Users\hreza87
```

String index of practise

```
string message = "Filter out stuff between *Howdy Stranger*";
bool validcheck=false;
while (validcheck==false)
{

    for(int i=0;i<message.Length;i++)
    {
        if (message[i]=='*')

            {
            int location= message.IndexOf(message[i]);
            Console.WriteLine(location);
            message = message.Remove(location,1);
            }

    }

}
```

This program uses nested loop, however since it deletes a *, the index of next one is one less then the supposed one . Hence watch next program which does not use index of but standard reference to i.

```
721  string message = "Filter out stuff between *Howdy Stranger*";
722
723  for(int i=0;i<message.Length;i++)
724  {
725      if (message[i]=='*')
726
727          {
728          int location=i;
729          Console.WriteLine(location);
730          }
731
732  }
```

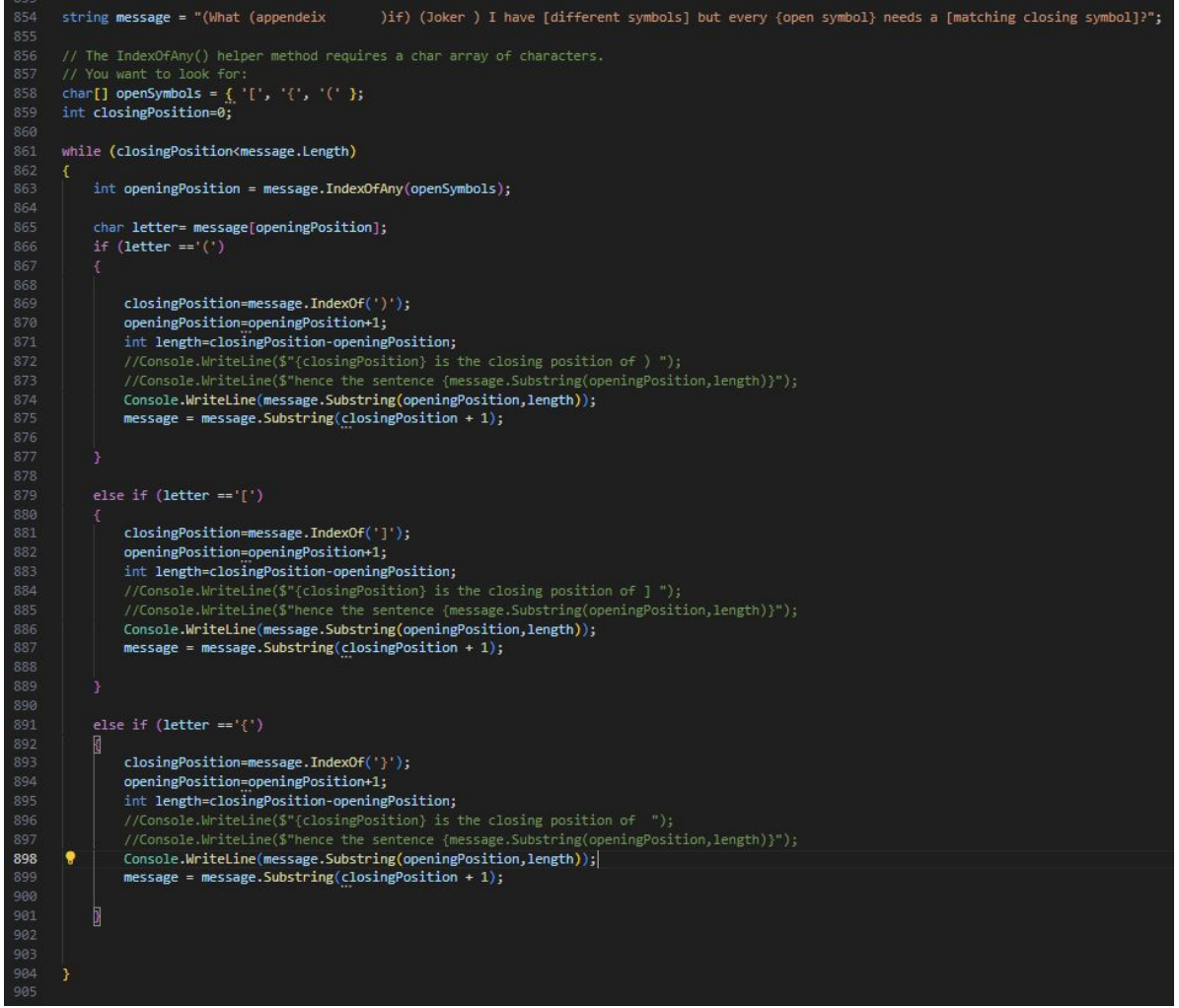
Using IndexOfAny() and LastIndexOf() helper methods

```
789   string message = "Help (find) the {opening symbols}";
790   Console.WriteLine($"Searching THIS Message: {message}");
791   char[] openSymbols = { '[', '{', '(' };
792   int startPosition = 5;
793   int openingPosition = message.IndexOfAny(openSymbols);
794   Console.WriteLine($"Found WITHOUT using startPosition: {message.Substring(openingPosition)}");
795
796   openingPosition = message.IndexOfAny(openSymbols, startPosition);
797   Console.WriteLine($"Found WITH using startPosition {startPosition}:  {message.Substring(openingPosition)}");
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\Module4\CsharpProjects\TestProject> dotnet run
Searching THIS Message: Help (find) the {opening symbols}
Found WITHOUT using startPosition: (find) the {opening symbols}
Found WITH using startPosition 5:  (find) the {opening symbols}
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\Module4\CsharpProjects\TestProject>
```

Some practise.



```
string message = "(What (appendeix       )if) (Joker ) I have [different symbols] but every {open symbol} needs a [matching closing symbol]?";

// The IndexOfAny() helper method requires a char array of characters.
// You want to look for:
char[] openSymbols = { '[', '{', '(' };
int closingPosition=0;

while (closingPosition<message.Length)
{
    int openingPosition = message.IndexOfAny(openSymbols);

    char letter= message[openingPosition];
    if (letter =='(')
    {

        closingPosition=message.IndexOf(')');
        openingPosition=openingPosition+1;
        int length=closingPosition-openingPosition;
        //Console.WriteLine($"{closingPosition} is the closing position of ) ");
        //Console.WriteLine($"hence the sentence {message.Substring(openingPosition,length)}");
        Console.WriteLine(message.Substring(openingPosition,length));
        message = message.Substring(closingPosition + 1);

    }

    else if (letter =='[')
    {
        closingPosition=message.IndexOf(']');
        openingPosition=openingPosition+1;
        int length=closingPosition-openingPosition;
        //Console.WriteLine($"{closingPosition} is the closing position of ] ");
        //Console.WriteLine($"hence the sentence {message.Substring(openingPosition,length)}");
        Console.WriteLine(message.Substring(openingPosition,length));
        message = message.Substring(closingPosition + 1);

    }

    else if (letter =='{')
    {
        closingPosition=message.IndexOf('}');
        openingPosition=openingPosition+1;
        int length=closingPosition-openingPosition;
        //Console.WriteLine($"{closingPosition} is the closing position of  ");
        //Console.WriteLine($"hence the sentence {message.Substring(openingPosition,length)}");
        Console.WriteLine(message.Substring(openingPosition,length));
        message = message.Substring(closingPosition + 1);

    }


}
```

string.IndexOfAny()

message.IndexOfAny(symbol,searching from certain index, ifits 0 it will start looking for any symbol from that point onwards. At the end of this program, that 0 eventually increasing, ignoring previous string that is being over written.

```
string message = "(What if) I have [different symbols] but every {open symbol} needs a [matching closing symbol]";

// The IndexOfAny() helper method requires a char array of characters.
// You want to look for:

char[] openSymbols = { '[', '{', '(' };




int closingPosition = 0;
while (true)
{
    int openingPosition = message.IndexOfAny(openSymbols,closingPosition);

    if (openingPosition == -1) break;

    string currentSymbol = message.Substring(openingPosition, 1);

    // Now  find the matching closing symbol
    char matchingSymbol = ' ';

    switch (currentSymbol)
    {
        case "[":
            matchingSymbol = ']';
            break;
        case "{":
            matchingSymbol = '}';
            break;
        case "(":
            matchingSymbol = ')';
            break;
    }
    // To find the closingPosition, use an overload of the IndexOf method to specify
    // that the search for the matchingSymbol should start at the openingPosition in the string.
    openingPosition += 1;
    closingPosition = message.IndexOf(matchingSymbol, openingPosition);

    // Finally, use the techniques you've already learned to display the sub-string:

    int length = closingPosition - openingPosition;
    Console.WriteLine(message.Substring(openingPosition, length));
}
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\Module4\CsharpProjects\TestProject> dotnet run
What if
different symbols
open symbol
matching closing symbol
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\Module4\CsharpProjects\TestProject>
```

Remove() and Replace()

```
963   string data = "12345John Smith         5000  3  ";
964   string updatedData = data.Remove(5, 20);
965   Console.WriteLine(updatedData);
966
967   string message = "This--is--ex-amp-le--da-ta";
968   message = message.Replace("--", " ");
969   message = message.Replace("-", "");
970   Console.WriteLine(message);
```

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\Module4\CsharpProjects\TestProject> dotnet run
123455000  3
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\Module4\CsharpProjects\TestProject> dotnet run
123455000  3
This is example data
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\Module4\CsharpProjects\TestProject>
```

Challenge replace and removing data.

```
string message = "What is the value <span>between the tags</span>?";

const string openSpan = "<span>";
const string closeSpan = "</span>";

int openingPosition = message.IndexOf(openSpan);
int closingPosition = message.IndexOf(closeSpan);

openingPosition += openSpan.Length;
int length = closingPosition - openingPosition;
Console.WriteLine(message.Substring(openingPosition, length));
```

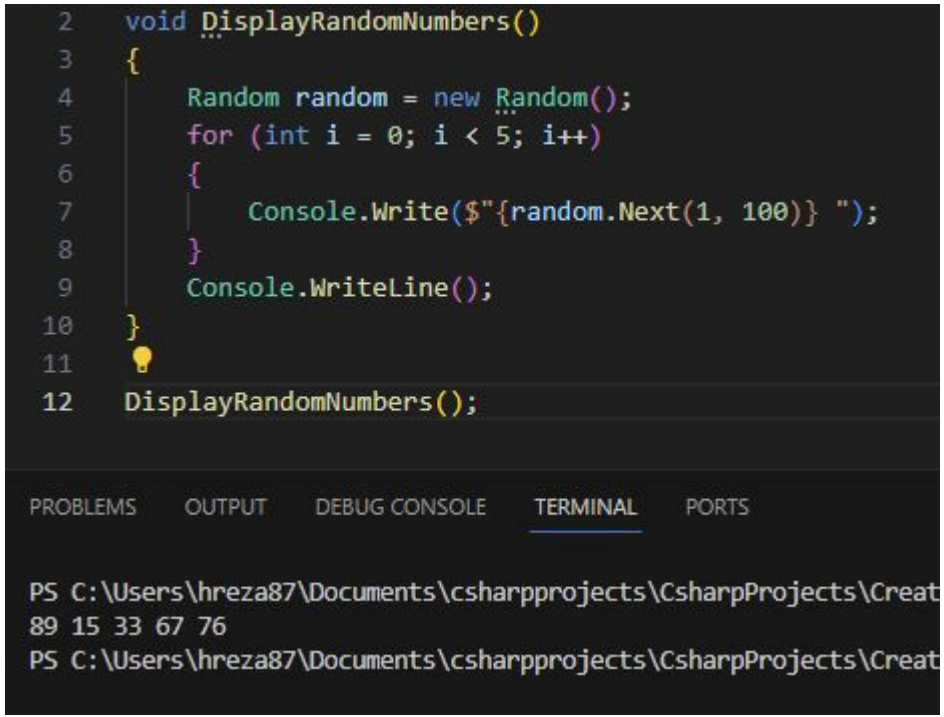
Create Methods in C# Console Applications

…void.. Methodname()

{

methodbody

}



```
string[] ipv4Input = {"107.31.1.5", "255.0.0.255", "555..0.555", "255...255"};
string[] address;
bool validLength = false;
bool validZeroes = false;
bool validRange = false;


foreach(string ip in ipv4Input)
{
    address = ip.Split(".", StringSplitOptions.RemoveEmptyEntries);
    ValidateLength();
    ValidateZeroes();
    ValidateRange();

    if (validLength && validZeroes && validRange)
        {
        Console.WriteLine($"ip is a valid IPv4 address");
        }
    else
        {
        Console.WriteLine($"ip is an invalid IPv4 address");
        }
}



void ValidateLength()
{
    validLength = address.Length == 4;
}
void ValidateZeroes()
{

    foreach (string number in address)
    {
        if (number.Length > 1 && number.StartsWith("0"))
        {
            validZeroes = false;
        }
    }
    validZeroes = true;
}
void ValidateRange()
{
    foreach (string number in address)
    {
        int value = int.Parse(number);
        if (value < 0 || value > 255)
        {
            validRange = false;
            return;
        }
    }
    validRange = true;

}
```

```
PROBLEMS     OUTPUT     DEBUG CONSOLE

PS C:\Users\hreza87\Documents\csharp
ip is a valid IPv4 address
ip is a valid IPv4 address
ip is an invalid IPv4 address
ip is an invalid IPv4 address
PS C:\Users\hreza87\Documents\csharp
```

IP address validation Program

Fortune teller app

```
196  Random random = new Random();
197  int luck = random.Next(100);
198
199  string[] text = {"You have much to", "Today is a day to", "Whatever work you do", "This is an ideal time to"};
200  string[] good = {"look forward to.", "try new things!", "is likely to succeed.", "accomplish your dreams!"};
201  string[] bad = {"fear.", "avoid major decisions.", "may have unexpected outcomes.", "re-evaluate your life."};
202  string[] neutral = {"appreciate.", "enjoy time with friends.", "should align with your values.", "get in tune with nature."};
203  TellFortune();
204  TellFortune();
205  luck = random.Next(100);
206  TellFortune();
207
208  void TellFortune()
209  {
210      Console.WriteLine("A fortune teller whispers the following words:");
211      string[] fortune = (luck > 75 ? good : (luck < 25 ? bad : neutral));
212      for (int i = 0; i < 4; i++)
213      {
214          Console.Write($"{text[i]} {fortune[i]} ");
215      }
216  }
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\CreatingMethods\CsharpProjects\TestProject> dotnet run
A fortune teller whispers the following words:
You have much to appreciate. Today is a day to enjoy time with friends. Whatever work you do should align with your values. This is an ideal time to ge
whispers the following words:
You have much to appreciate. Today is a day to enjoy time with friends. Whatever work you do should align with your values. This is an ideal time to ge
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\CreatingMethods\CsharpProjects\TestProject> dotnet run
A fortune teller whispers the following words:
You have much to fear. Today is a day to avoid major decisions. Whatever work you do may have unexpected outcomes. This is an ideal time to re-evaluate
e following words:
You have much to appreciate. Today is a day to enjoy time with friends. Whatever work you do should align with your values. This is an ideal time to ge
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\CreatingMethods\CsharpProjects\TestProject> dotnet run
A fortune teller whispers the following words:
You have much to look forward to. Today is a day to try new things! Whatever work you do is likely to succeed. This is an ideal time to accomplish your
ollowing words:
You have much to look forward to. Today is a day to try new things! Whatever work you do is likely to succeed. This is an ideal time to accomplish your
ollowing words:
You have much to appreciate. Today is a day to enjoy time with friends. Whatever work you do should align with your values. This is an ideal time to ge
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\CreatingMethods\CsharpProjects\TestProject>
```

Methods with parameters

```
220  void DisplayAdjustedTimes(int[] times, int currentGMT, int newGMT)
221  {
222      int diff = 0;
223      if (Math.Abs(newGMT) > 12 || Math.Abs(currentGMT) > 12)
224      {
225          Console.WriteLine("Invalid GMT");
226      }
227
228      else if (newGMT <= 0 && currentGMT <= 0 || newGMT >= 0 && currentGMT >= 0)
229      {
230          diff = 100 * (Math.Abs(newGMT) - Math.Abs(currentGMT));
231      }
232      else
233      {
234          diff = 100 * (Math.Abs(newGMT) + Math.Abs(currentGMT));
235      }
236
237      for (int i = 0; i < times.Length; i++)
238  {
239      int newTime = ((times[i] + diff)) % 2400;
240      Console.WriteLine($"{times[i]} -> {newTime}");
241  }
242  }
243
244
245  int[] schedule = { 800, 1200, 1600, 2000 };
246  DisplayAdjustedTimes(schedule, 2, 12);
247
248
249
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\CreatingMethods\CsharpProjects\TestP
800 -> 1800
1200 -> 2200
1600 -> 200
2000 -> 600
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\CreatingMethods\CsharpProjects\TestP
```

C# Methods

```
class Program
{
  static void MyMethod()
  {
    // code to be executed
  }
}
```

The static means this method is part of the program class and no other objects that derive from it.

Void means there is no return type for this, it probability prints stuff using console.WriteLine.

```
using System;

namespace MyApplication
{
  class Program
  {
    static void MyMethod(string child1, string child2, string child3)
    {
      Console.WriteLine("The youngest child is: " + child3);
    }

    static void Main(string[] args)
    {
      MyMethod(child3: "John", child1: "Liam", child2: "Liam");
    }
  }
}
```

Named arguments can be given as parameters using key:value syntax.

As longas you refer to key and value you wont cause issues.

```
The youngest child is: John
```

```
using System;

namespace MyApplication
{
    0 references
    class Program
    {
        1 reference
        static void MyMethod()
        {
            Console.WriteLine("I just got executed!");
        }

        1 reference
        static void MyMethod(string fname, int age)
        {
            Console.WriteLine(fname + " is " + age);
        }

        0 references
        static void Main(string[] args)
        {
            MyMethod();
            MyMethod("Jenny", 8);
        }
    }
}
```

Using system facilitates we can use libraries classes from that given class.
Name space is a container of classes , can store more then one apparently.

In this example we can overload a single method with multiple argument signatures. They will do stuff differently.

C# Classes

```
using System;
namespace MyApplication
{
    1 reference
    class Car
    {
        1 reference
        static string color = "red";

        0 references
        static void Main(string[] args)
        {
            //Car myObj = new Car();
            Console.WriteLine(Car.color);
        }
    }
}
```

Static ensures that method or data belongs to the class and not object.

This attribute belongs to class, when i do myObj.color that would not work, that is why i simply refer to the class name below.
Car.color will display the variable.

```
using System;
namespace MyApplication
{
    4 references
    class Car
    {
        2 references
        public string color = "red";
    }

    0 references
    class Program
    {
        0 references
        static void Main(string[] args)
        {
            Car myObj1 = new Car();
            Car myObj2 = new Car();
            Console.WriteLine(myObj1.color);
            Console.WriteLine(myObj2.color);
        }

    }
}
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

PS C:\Users\hreza87\Documents\csharpprojects\Cs
red
red
PS C:\Users\hreza87\Documents\csharpprojects\Cs
```

Public in public string colour ensures that this variable is accessible by other classes such as program.

If i had put static then if i had created new objects, and referred to the variable defined in class i would not have been able to. Hence i got rid of static.

```
using System;

namespace MyApplication
{
  class Car
  {
    string color = "red";
    string color2 = "Purple";
    string sentence(string word)//method created
        {
        word=word +" is a good colour";
        return word;
        }

    static void Main(string[] args)
    {
      Car myObj = new Car();
      Console.WriteLine(myObj.color);
      Console.WriteLine(myObj.color2);
      Console.WriteLine(myObj.sentence("Orange"));
    }
  }
}
```

```
red
Purple
Orange is a good colour
```

```
//filename: Car.cs
using System;

namespace MyApplication
{
  class Car
  {
    string model;
    string color;
    int year;

    static void Main(string[] args)
    {
      Car Ford = new Car();
      Ford.model = "Mustang";
      Ford.color = "red";
      Ford.year = 1969;

      Car Opel = new Car();
      Opel.model = "Astra";
      Opel.color = "white";
      Opel.year = 2005;

      Console.WriteLine(Ford.model);
      Console.WriteLine(Opel.model);
    }
  }
}
```

```
Mustang
Astra
```

Multiple Classes accessing class members from one another

```
using System;
using Microsoft.VisualBasic;

namespace MyApplication
{
    4 references
    class Car
    {
        4 references
        public string? model;
        2 references
        public string? color;
        2 references
        public int year;
        1 reference
        public void fullThrottle()
        {
            Console.WriteLine("The car is going as fast as it can!");
        }
    }
    0 references
    class Program
    {
        0 references
        static void Main(string[] args)
        {
            Car Ford = new Car();
            Ford.model = "Mustang";
            Ford.color = "red";
            Ford.year = 1969;

            Car Opel = new Car();
            Opel.model = "Astra";
            Opel.color = "white";
            Opel.year = 2005;

            Console.WriteLine(Ford.model);
            Console.WriteLine(Opel.model);
            Opel.fullThrottle();
        }
    }
}
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\CreatingMethods\CsharpPr
Mustang
Astra
The car is going as fast as it can!
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\CreatingMethods\CsharpPr
```

Using a Constructor in a class with parameters.

Allows when creating new objects to assign values.

Solves the problem of null values.

```
using System;
using Microsoft.VisualBasic;

namespace MyApplication
{
    5 references
    class Car
    {
        3 references
        public string model;
        3 references
        public string color;
        3 references
        public int year;

        2 references
        public Car(string modelName, string modelColor, int modelYear)
        {
            model = modelName;
            color = modelColor;
            year = modelYear;
        }
    }

    0 references
    class Program
    {
        0 references
        static void Main(string[] args)
        {
            Car Ford = new Car("Mustang", "Red", 1969);
            Car Audi = new Car("Audi", "Grey", 2017);
            Console.WriteLine(Ford.color + " " + Ford.year + " " + Ford.model);
            Console.WriteLine(Audi.color + " " + Audi.year + " " + Audi.model);
        }
    }
}
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\CreatingMethods\CsharpProjects\TestProject> dotnet run
Red 1969 Mustang
Grey 2017 Audi
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\CreatingMethods\CsharpProjects\TestProject> dotnet run
Red 1969 Mustang
Grey 2017 Audi
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\CreatingMethods\CsharpProjects\TestProject>
```

Constructors Save Time

When you consider the example from the previous chapter, you will notice that constructors are very useful, as they help reducing the amount of code:

Without constructor:

prog.cs

```
class Program
{
  static void Main(string[] args)
  {
    Car Ford = new Car();
    Ford.model = "Mustang";
    Ford.color = "red";
    Ford.year = 1969;

    Car Opel = new Car();
    Opel.model = "Astra";
    Opel.color = "white";
    Opel.year = 2005;
```

With constructor:

prog.cs

```
class Program
{
  static void Main(string[] args)
  {
    Car Ford = new Car("Mustang", "Red", 1969);
    Car Opel = new Car("Astra", "White", 2005);

    Console.WriteLine(Ford.model);
    Console.WriteLine(Opel.model);
  }
}
```

C# Properties (Get and Set)

Encapsulation

..model variable is private.

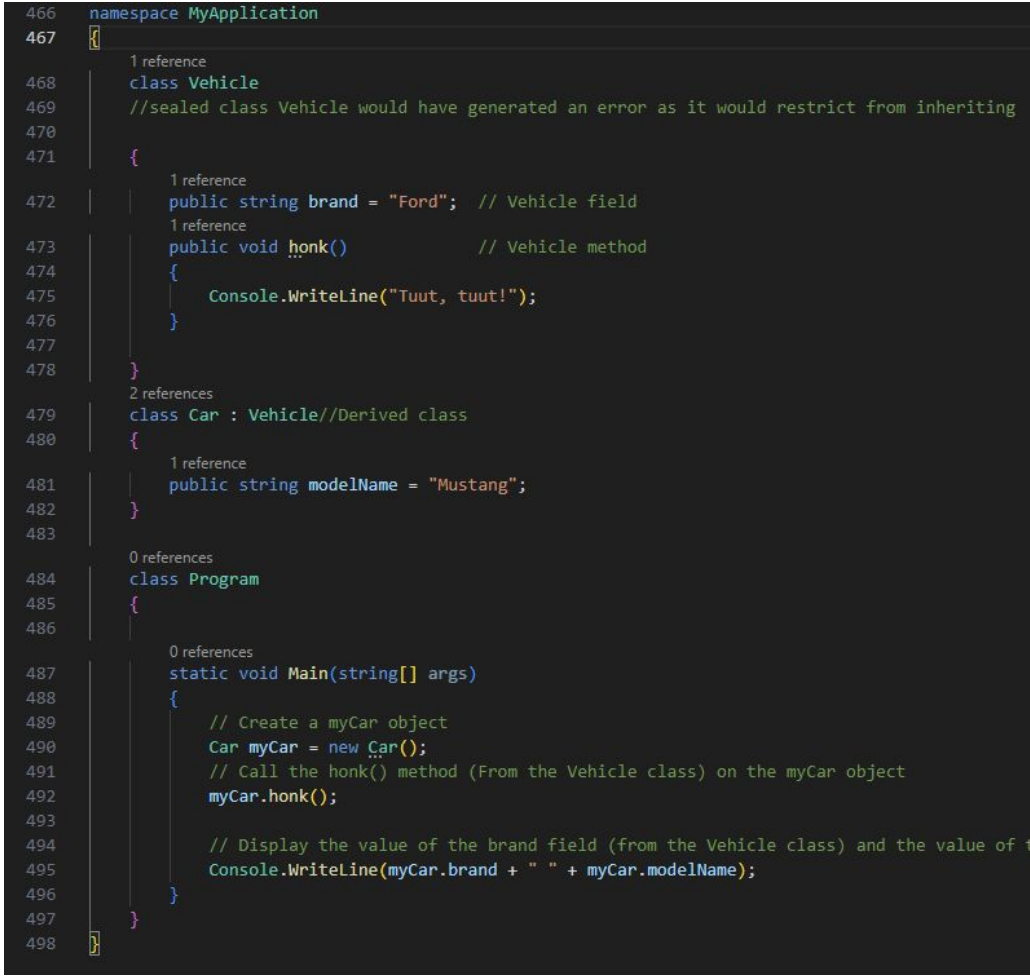
A public property has a method which provides access to the variable model and also mutation to change the value without directly accessing it.

```
using System;
using System.Reflection.Metadata.Ecma335;
using Microsoft.VisualBasic;

namespace MyApplication
{
    5 references
    class Car
    {
        3 references
        private string model;

        1 reference
        public string Model
        {
            get{return model;} //get method
            set{model=value;} // set method
        }

        2 references
        public string color;
        2 references
        public int year;

        2 references
        public Car(string modelName, string modelColor, int modelYear)
        {
            model = modelName;
            color = modelColor;
            year = modelYear;
        }
    }

    0 references
    class Program
    {
        0 references
        static void Main(string[] args)
        {
            Car Ford = new Car("Mustang", "Red", 1969);
            Car Audi = new Car("Audi", "Grey", 2017);
            Console.WriteLine(Ford.color + " " + Ford.year + " " + Ford.Model);
            //Console.WriteLine(Audi.color + " " + Audi.year + " " + Audi.model);
        }
    }
}
```

C# Inheritance (Derived and class)

Child class: parent class

Child class takes all variable sand methods from parent class.



```
namespace MyApplication
{
    1 reference
    class Vehicle
    //sealed class Vehicle would have generated an error as it would restrict from inheriting

    {
        1 reference
        public string brand = "Ford";  // Vehicle field
        1 reference
        public void honk()             // Vehicle method
        {
            Console.WriteLine("Tuut, tuut!");
        }

    }
    2 references
    class Car : Vehicle//Derived class
    {
        1 reference
        public string modelName = "Mustang";
    }

    0 references
    class Program
    {

        0 references
        static void Main(string[] args)
        {
            // Create a myCar object
            Car myCar = new Car();
            // Call the honk() method (From the Vehicle class) on the myCar object
            myCar.honk();

            // Display the value of the brand field (from the Vehicle class) and the value of
            Console.WriteLine(myCar.brand + " " + myCar.modelName);
        }
    }
}
```

PolyMorphism

In base class declare method as virtual

In derived class declare as override so that when new objects are constructed and method is called it mimics behaviour of the newly created object as opposed to the base class.

```
502  using System;
503
504  namespace MyApplication
505  {
         6 references
506      class Animal  // Base class (parent)
507      {
             5 references
508          public virtual void animalSound()
509          {
510              Console.WriteLine("The animal makes a sound");
511          }
512      }
513
         1 reference
514      class Pig : Animal  // Derived class (child)
515      {
             4 references
516          public override void animalSound()
517          {
518              Console.WriteLine("The pig says: wee wee");
519          }
520      }
521
         1 reference
522      class Dog : Animal  // Derived class (child)
523      {
             4 references
524          public override void animalSound()
525          {
526              Console.WriteLine("The dog says: bow wow");
527          }
528      }
529
         0 references
530      class Program
531      {
             0 references
532          static void Main(string[] args)
533          {
534              Animal myAnimal = new Animal();  // Create a Animal object
535              Animal myPig = new Pig();  // Create a Pig object
536              Animal myDog = new Dog();  // Create a Dog object
537
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\CreatingMethods\CsharpProjects\TestProject> dotnet run
Tuut, tuut!
Ford Mustang
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\CreatingMethods\CsharpProjects\TestProject> dotnet run
The animal makes a sound
The pig says: wee wee
The dog says: bow wow
PS C:\Users\hreza87\Documents\csharpprojects\CsharpProjects\CreatingMethods\CsharpProjects\TestProject>
```

C# Abstraction

To prevent base class to create objects, or use it's methods, it must be derived into a child class, as shown right.

```
using System;

namespace MyApplication
{
  // Abstract class
  abstract class Animal
  {
    // Abstract method (does not have a body)
    public abstract void animalSound();
    // Regular method
    public void sleep()
    {
      Console.WriteLine("Zzz");
    }
  }

  // Derived class (inherit from Animal)
  class Pig : Animal
  {
    public override void animalSound()
    {
      // The body of animalSound() is provided here
      Console.WriteLine("The pig says: wee wee");
    }
  }

  class Program
  {
    static void Main(string[] args)
    {
      Pig myPig = new Pig();  // Create a Pig object
      myPig.animalSound();
      myPig.sleep();
    }
  }
}
```